MULTIMEDIA UNIVERSITY

STUDENT ID NO

MULTIMEDIA UNIVERSITY

FINAL EXAMINATION

TRIMESTER 1, 2017/2018

TCP1311 – OBJECT ORIENTED PROGRAMMING

(All sections / Groups)

11 October 2017
9 AM – 11 AM
(2 Hours)

INSTRUCTIONS TO STUDENTS

1. Answer **ALL** questions
2. This question paper has 5 printed pages excluding the front cover
3. Please print all your answers in the answer booklet provided

QUESTION 1

(a) What is encapsulation and why it is important? Demonstrate a code example of encapsulation using Java language.

[10 marks]

(b) A student has got a name, age, and CGPA. The student can also enroll in many courses. Before enrolling a course, a check is done to ensure the student has not reached a maximum course a student can take.

For the scenario above, provide the Java code for class Student. Include necessary constructors, data members, and method implementations.

[11 marks]

(c) Describe the differences between a constructor and a method.

[4 marks]

Continued .....

QUESTION 2

(a) What is the difference between static and non-static methods? When should one use static or non-static method? Provide code samples to illustrate your points.

[9 marks]

(b) What is a Java interface? How it can be used to achieve multiple-inheritance? Show using code samples how multiple-inheritance can be achieved.

[10 marks]

(c) Given the following programs. Write the output produced by the following programs.

```
public class Employee {
        private String name;
        private String address;
        private int number;
        public Employee(String name, String address, int number) {
                System.out.println("Constructing an Employee");
                this.name = name;
                this.address = address;
                this.number = number;
        }
        public void mailCheck() {
                System.out.println("Mailing a check to " + this.name + " " +
                        this.address);
        }
        public String toString() {
                return name + " " + address + " " + number;
        }
        public String getName() {
                return name;
        }
        public String getAddress() {
                return address;
        }
        public void setAddress(String newAddress) {
                address = newAddress;
        }
        public int getNumber() {
                return number;
        }
}
```

Continued .....

```
public class Salary extends Employee {
        private double salary; // Annual salary
        public Salary(String name, String address, int number, double salary) {
        super(name, address, number);
                setSalary(salary);
        }

        public void mailCheck() {
                System.out.println("Within mailCheck of Salary class ");
                System.out.println("Mailing check to " + getName() + " with salary "
                        + salary);
        }
        public double getSalary() {
                return salary;
        }
        public void setSalary(double newSalary) {
                if(newSalary >= 0.0) {
                        salary = newSalary;
                }
        }
        public double computePay() {
                System.out.println("Computing salary pay for " + getName());
                return salary/52;
        }
}
public class Demo {
        public static void main(String [] args) {
                Salary s = new Salary("Mohd Mohtashim", "Ambehta, UP", 3,
                        3600.00);
                Employee e = new Salary("John Adams", "Boston, MA", 2, 2400.00);
                s.mailCheck();
                e.mailCheck();
        }
}
```

[6 marks]

Continued .....

QUESTION 3

(a) A Customer will be able to add products to an order and then edit them before purchase. There is also an inventory for the products and the inventory details can be modified.

Design a complete UML class diagram for the scenario above with necessary data members and methods along with class relationships.

[10 marks]

(b) Draw a sequence diagram for the following sequence of interactions.

i. The message *makePayment* is sent to an instance of a *Register*. The sender is not identified.

ii. The *Register* instance sends the *makePayment* message to a *Sale* instance.

iii. The *Sale* instance creates an instance of a *Payment*.

Then, develop a code fragment for the *Sale* class based on the interaction above.

[10 marks]

(c) For the given code below, draw a class diagram.

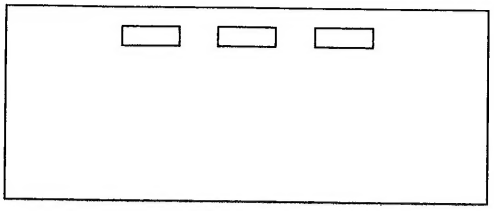
```
public class Employee {
        private static String department = "R&D";
        private int empId;
        private Employee(int employeeId) {
                this.empId = employeeId;
        }
        public static String getEmployee(int emplId) {
                if (emplId == 1) {
                        return "idiotechie";
                } else {
                        return "Employee not found";
                 }
         }
        public static String getDepartment() {
                return department;
        }
}
```

[5 marks]

Continued .....

QUESTION 4

(a) Provide a source code that will create a flow layout for the 3 buttons as follows:



[5 marks]

(b) Identify and explain the error(s) in the following code. Then correct the error(s).

```
public int readint(){
        int input;
        input = System.in.read();
       return input;
}
```

[10 marks]

(c) Provide a Java code example of decorator design pattern application. Also, draw an UML diagram to illustrate the decorator design pattern.

[10 marks]

End of Page